23 Feb The Fast Fourier Transform

Announcements:

1. No office hours Feb break (Sat – Tues)
2. Prob Set 2 grades to be released today
   Prelim grades ASAP.

Multiply polynomials

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$
$$B(x) = b_0 + \cdots + b_{n-1} x^{n-1}$$

Find the coefficients of the product.

Given sequences $\underbrace{(a_0, \ldots, a_{n-1})}_{=\mathbf{a}}$ and $\underbrace{(b_0, \ldots, b_{n-1})}_{=\mathbf{b}}$

find their convolution $\mathbf{c} = (c_0, \ldots, c_{2n-2})$

given by

$$c_k = \sum_{i+j=k} a_i b_j$$

A proposal for multiplying polynomials fast.

0. Choose points $z_0, \ldots, z_{2n-1}$ where we plan to evaluate the polynomials.

$O(n^2) \longrightarrow$ 1. Calculate $A(z_i)$ and $B(z_i)$ for $i = 0, \ldots, 2n-1$.

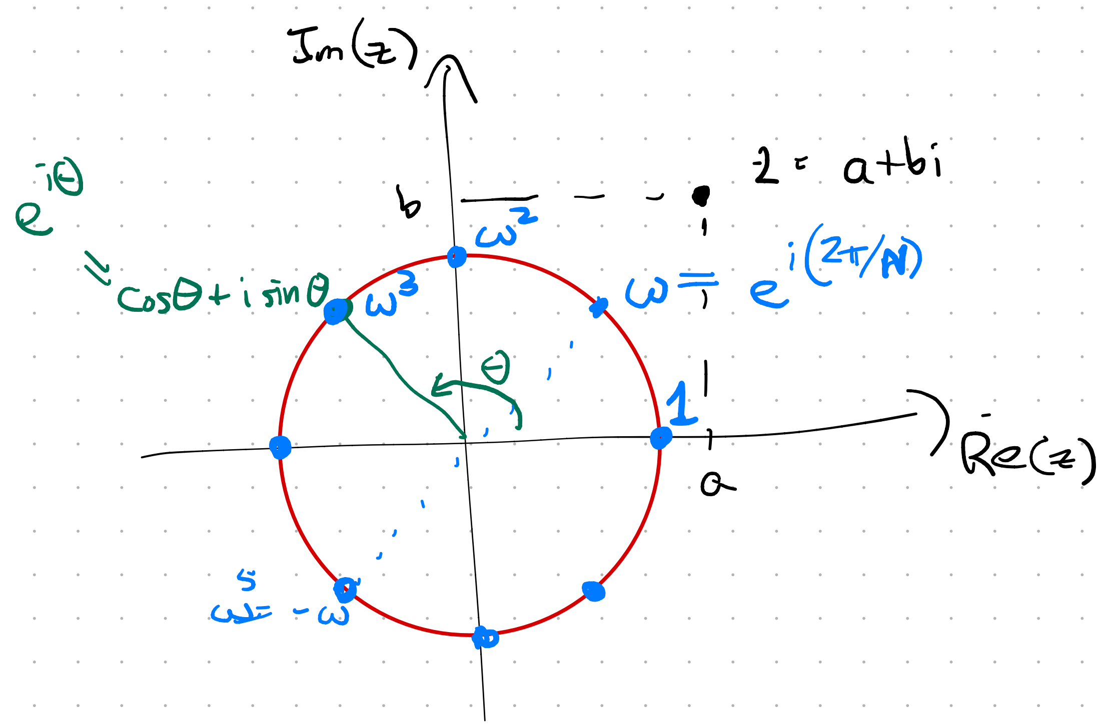
$O(n) \longrightarrow$ 2. Calculate $C(z_i) = A(z_i) \cdot B(z_i)$ for $i = 0, \ldots, 2n-1$.

3. "Interpolation": Find the coefficients of the unique degree $2n-1$ polynomial $C(x)$ taking the values calculated in step 2, at $z_0, \ldots, z_{2n-1}$.

The Fast Fourier Transform. An $O(n \log n)$ algorithm to do steps 1 & 3, when $z_0, \ldots, z_{2n-1}$ are the complex $2n^{th}$ roots of unity.

$2n = N$

$\omega = e^{\frac{2\pi i}{N}}$ is a "primitive n^{th} root of unity"



$$\omega^N = e^{2\pi i} = \cos(2\pi) + i\sin(2\pi) = 1.$$

If $j \in \mathbb{Z}$

$$(\omega^j)^N = (\omega^N)^j = 1.$$

The (discrete) Fourier transform of a sequence $a_0, \ldots, a_{N-1}$ is the sequence $\hat{a} = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{N-1})$ where

$$\hat{a}_j = A(\omega^j) = a_0 + a_1\omega^j + a_2\omega^{2j} + \cdots + a_{N-1}\omega^{(N-1)j}$$

Computing $FT(a_0, \ldots, a_{n-1})$ is the same as evaluating $A(\omega^j)$ for $j = 0, \ldots, N-1$.

The DFT matrix of order N is

$$\begin{array}{c} \\ j \end{array} \begin{bmatrix} & & i & & \\ & & \vdots & & \\ - & - & \omega^{ij} & & \\ & & & & \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \end{bmatrix} = \begin{bmatrix} \\ \sum a_i \omega^{ij} \\ \\ \end{bmatrix} \begin{array}{c} = \hat{a} \\ \leftarrow j \end{array}$$

$\| F$

$$F^2 = \begin{bmatrix} N & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & & N \\ 0 & 0 & \cdots & N & \\ 0 & 0 & N & & \\ 0 & N & & & \end{bmatrix}$$

Step 3 of the polynomial multiplication algorithm reduces to Step 1 because the identity

$$F^2 = \begin{bmatrix} N & 0 & 0 & 0 \\ 0 & 0 & \cdots & N \\ \vdots & & N^{\cdot^{\cdot}} & \\ 0 & N & & 0 \end{bmatrix}$$

shows that the inverse Fourier transform (Step 3) is the Fourier transform (Step 1) followed by scaling by $\frac{1}{N}$ and permuting coordinates (both linear time operations).

Write $A(x) = A_{even}(x^2) + x \cdot A_{odd}(x^2)$

$$A_{even}(y) = a_0 + a_2 y + a_4 y^2 + \cdots + a_{N-2}\, y^{\frac{N}{2}-1}$$

$$A_{odd}(y) = a_1 + a_3 y + a_5 y^2 + \cdots + a_{N-1}\, y^{\frac{N}{2}-1}$$

To evaluate $A(\omega^j)$ for $j = 0, \ldots, N-1$:

FT of size $\frac{N}{2}$:

1. Eval $A_{even}(\omega^{2j})$ for $j = 0, \ldots, \frac{N}{2}-1$
2. Eval $A_{odd}(\omega^{2j})$ for $j = 0, \ldots, \frac{N}{2}-1$
3. Compute $A(\omega^j) = A_{even}(\omega^{2j}) + \omega^j \cdot A_{odd}(\omega^{2j})$.

Assume $N = 2^k$.

$$T(N) = 2\,T\left(\frac{N}{2}\right) + O(N)$$

$$\Longrightarrow \quad T(N) = O(N \log N).$$